

Lab 2

Deadline: Feb 7, 2024

You are tasked with implementing a program that allows a user to dynamically sort a list of integers based on their preferred sorting order. The user should be able to specify whether the sorting should be in ascending or descending order.


You are provided with a skeleton file, you are not allowed to modify any other part of code except **selection_sort()** function from line 16.

```
main.py
1  #012345
2  import sys
3
4  def selection_sort(arr, ascending=True):
5      """
6      Perform Selection Sort on the given list.
7
8      Parameters:
9      - arr (list): The list to be sorted.
10     - ascending (bool): If True, sort in ascending order; otherwise, sort in descending
11
12     Returns:
13     - list: The sorted list.
14     """
15     #WRITE YOUR CODE otherwise
16
17     #DO NOT WRITE ANYTHING BELOW THIS
18     return arr
19
20
21 def main():
22     # Check for the correct number of command line arguments
23     if len(sys.argv) != 2:
```

Requirements

1. Input

- ◆ The skeleton file contains an initialized list of 10 numbers.
- ◆ A command line argument is used to specify the sorting order e.g 'A' for ascending and 'D' for descending.

input

Command line arguments:

Standard Input: ☒ Interactive Console ☐ Text

2. Order Preference

- ◆ Allow the user to specify whether the sorting should be in ascending or descending order.

3. Sorting

- ◆ Implement the simple selection sort algorithm to sort the list based on the user's preferences.
- ◆ Example: Let first element as minimum or maximum, loop through the list and compare it with next element.

4. Output

- ◆ The program should only print the sorted list which is already defined in the skeleton file, you are not allowed to write any print statement in the code.

5. Save the File Offline

- ◆ Click the "Download Code" button/icon to save the file offline.

6. Submission

- ◆ Include your Student ID as comment at the top of your code i.e #012345. Rename the main.py file to lab2.py. Place the lab2.py in a folder and compress to a zip file. Submit it to moodle.

7. Grading

- ◆ (1 point) Correct submission of working code with no errors and loop implementation.
- ◆ (5 point) Correct solution to the problem.